Operations Research Letters 37 (2009) 312–316

Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

Instability of FIFO in a simple queueing system with arbitrarily low loads

Tolga Tezcan

Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, 117 Transportation Bldg, Urbana, IL 61801, United States

ARTICLE INFO

Article history:
Received 21 October 2008
Accepted 6 May 2009
Available online 23 May 2009

Keywords:
FIFO
Queueing systems
Stability
Fluid models

ABSTRACT

We show, using a simple example, that the First-In-First-Out (FIFO) policy can be unstable in a system with arbitrarily low load. Our proof is based on the observation that the special structure of the example we use allows us to establish stability using a much simpler queueing system.

Published by Elsevier B.V.

1. Introduction

It is now well known that a system that has theoretically sufficient capacity to meet all the demand is not necessarily stable [1,2]. There is a vast literature on how to establish stability of queueing systems using fluid modes or drift rate arguments; see [3,4]. These tools have been successfully used to establish stability of queueing systems in many different settings. Stability of FIFO has also been studied extensively; see [5–7] and the references therein. It has been shown in [5] for a stochastic queueing network and in [7] for an adversarial queue that FIFO can be unstable in arbitrarily low loads. In this paper, we prove the instability of FIFO in arbitrarily low loads using a very simple example compared to those in [5,7].

Consider a queueing system that consists of two job classes and two server pools, see Fig. 1(a). We refer to these systems as X-systems. Let λ_j denote the arrival rate to class j. Let μ_{ij} denote the service rate of a class j job by server i for $i, j = 1, 2$. We assume that $\mu_{ij} > 0$ for all $i, j = 1, 2$. Also assume that service times and interarrival times are exponential. Define

$$\theta(\mu, \lambda) = \frac{\mu_{11}\mu_{12}}{\lambda_1\mu_{12} + \lambda_2\mu_{11}} + \frac{\mu_{21}\mu_{22}}{\lambda_1\mu_{22} + \lambda_2\mu_{21}}. \tag{1.1}$$

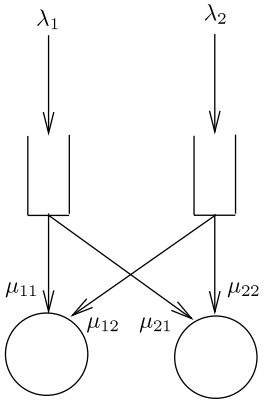
Our main result (Theorem 2.1) is that if $\theta(\mu, \lambda) > 1$ the system is positive Harris recurrent (see, [3]), i.e., the underlying Markov process has a stationary distribution, and it is rate stable if $\theta(\mu, \lambda) = 1$ and transient if $\theta(\mu, \lambda) < 1$. From (1.1), it is not difficult to see that FIFO can be unstable even when the load on the system is very small. For example, let $\lambda_1 = \lambda_2 = 1$, $\mu_{11} = \mu_{22} = 1000$ and $\mu_{21} = \mu_{12} = 0.1$ so that $\theta(\mu, \lambda) \approx 0.2$, hence the system is unstable. We first note that as long as server 1 is not allowed to serve class 2 jobs and server 2 is not allowed to serve class 1 jobs, the system will be stable and the utilization of the servers will be 0.01%. It is not difficult to see that the system will be unstable as long as $\mu_{21} < 0.5$ and $\mu_{12} < 0.5$ no matter how large μ_{11} and μ_{22} are. To illustrate, we simulate this system with $\mu_{21} = \mu_{12} = 0.49$ and $\mu_{21} = \mu_{12} = 0.51$ with $\mu_{11} = \mu_{22} = 1000$. The result are shown in Fig. 1(b). The increasing curve is the number of jobs in the first queue when service rate is $\mu_{21} = \mu_{12} = 0.49$ and the line on the bottom is for the system with $\mu_{21} = \mu_{12} = 0.51$. The number of jobs in the second queue, which is not plotted, exhibits a similar trend.

The idea of the proof of our main result is based on the fact that, when there are jobs in a queue waiting, the class of the job a server will serve next does not depend on the system history. For instance, in the numerical example we gave in the previous paragraph, the probability that the next job belongs to either class is 50% and this is independent from the state of the system. Given this fact, it is not difficult to see that the system is unstable, since the average time server j takes to finish service is $0.5/\mu_{j1} + 0.5/\mu_{j2}$. Again, if $\mu_{12} = \mu_{21} < 0.5$, this implies on average the service time of a job from either server is greater than 1. Since the total arrival rate is 2, the queue lengths are bound to explode. In general, the analysis of the stability of FIFO is very complicated. We prove the result by showing that the X-model under FIFO is equivalent to another system with a single queue. The stability of this new system can be established using traditional fluid models; see [8].

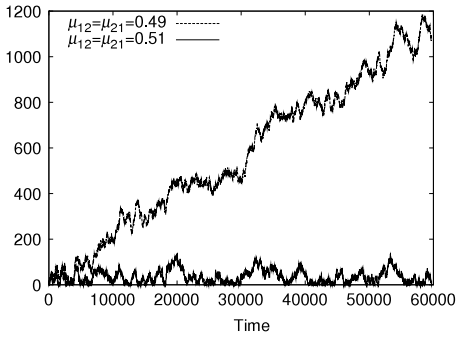
In [5], a similar result in a queuing system, where jobs visit each server several times and service time depends on the number of the visit, has been shown. However, unlike our example, the example used there has more and more servers as the load gets smaller and smaller. In [7], an adversarial queue in a system whose graph has to have at least a diameter $O(r^{-1}\log(1/r)^3)$ if the arrival

E-mail address: ttezcan@uiuc.edu.

0167-6377/$ – see front matter. Published by Elsevier B.V.
doi:10.1016/j.orl.2009.05.001



(a) An X-model queueing system.



(b) Simulation results for two systems.

Fig. 1. X-model and simulation results.

rate is r has been used to show instability of FIFO. Compared to these systems, our example is very simple. This is due to the fact that they only consider systems where each job class can only be served by a unique server. Allowing all servers to serve all the job classes simplifies the proof considerably. Such simplification is not possible, even when we add another server that can only serve one of the classes. However, it is possible to extend our result to the case with several servers and several job classes as long as all the servers can handle all the job classes.

It should be clear from our example above that the main reason why FIFO is not stable is because servers use activities, job class-server matchings, that are not very "efficient". It is common in the literature to devise scheduling policies assuming which activities are efficient; see [9,10] among others. This is accomplished by formulating a static planning problem (SPP) (see Section 2) and determining from its solution which activities should be used, called the basic activities, and which activities should not be used, called the non-basic activities. However, the SPP requires information about arrival rates, which is not always available. We note, however, that there are policies such as max-weight; see [11] and the references therein, that do not require the information about which activities are basic.

The rest of this paper is organized as follows. In Section 2 we present the details of the queueing system and our main result. In Section 3 we prove the main result.

2. Queuing model and main result

Consider the X-model introduced in the previous section. Assume that servers are dispatched according to a FIFO policy; when a server finishes service, that server starts serving the longest waiting job in the system if there are any. How a job is routed to servers when there is an arrival to an empty system does not matter. For concreteness, it can be assumed that a server is picked randomly with equal probability. Service times depend both on the class and the server providing service. We assume that interarrival times and service times are exponentially distributed. Let μ_{ij} denote the service rate of a class j job by server i for $i, j = 1, 2$. We assume that $\mu_{ij} > 0$ for all $i, j = 1, 2$. The arrival rate to class j is denoted by λ_j.

In order to define the load on the system, it is customary to formulate a linear program that is known as the static planning problem (SPP). The SPP in this setting is defined by

$$\min \rho$$

s.t.

$$\sum_{i=1}^{2} \mu_{ij} x_{ij} = \lambda_j, \quad \text{for } j = 1, 2,$$

$$\sum_{j=1}^{2} x_{ij} \le \rho, \quad \text{for } i = 1, 2,$$

$$x_{ij} \ge 0, \quad \text{for } j = 1, 2 \text{ and } i = 1, 2.$$

The quantity x_{ij} can be thought of as the long-run proportion of time server i serves class j jobs. The objective of the SPP is to minimize the nominal utilization of the busiest server pool. Let (ρ^*, x^*) be an optimal solution to the SPP. If $\rho^* \le 1$ then theoretically, the system can be made stable under some policy. Hence, we call ρ^* the load on the system. For the example we used in Section 1, the optimal solution to SPP is $\rho^* = 0.1\%$, $x^*_{11} = x^*_{22} = 0.001$, $x^*_{12} = x^*_{21} = 0$. To illustrate the main claim of this paper, let $\mu_{12} = \mu_{21}$, $\mu_{11} = \mu_{22} > \lambda_1 = \lambda_2 = 1$, and $\mu_{12} < \mu_{11}$, then $\rho^* = 1/\mu_{11}$. As discussed in Section 1, as long as $\mu_{21} < 0.5$, the system is unstable no matter what the value of μ_{11} is. Hence, although the system is unstable, the load on the X-model can be made arbitrarily low by choosing μ_{11} arbitrarily large.

Next, we give precise definitions of stability and instability before we present our main result. Let $Q_j(t)$ denote the number of class j jobs in queue and $Q(t) = (Q_1(t), Q_2(t))$. A queueing network is said to be *rate stable* if for each initial fixed data

$$\frac{\|Q(t)\|}{t} \Rightarrow 0 \quad \text{as } t \to \infty,$$

with probability one, where $\|\cdot\|$ is the max norm. A queueing system is said to be positive Harris recurrent if the underlying Markov process possesses a unique stationary distribution. A queueing system is said to be transient if the underlying Markov process does not admit a stationary distribution. Next we present our main result.

Theorem 2.1. *Consider an X-model system operating under FIFO.*

(a) *It is positive Harris recurrent if* $\theta(\mu, \lambda) > 1$.
(b) *It is rate stable if* $\theta(\mu, \lambda) \ge 1$.
(c) *It is transient if* $\theta(\mu, \lambda) < 1$.

Remark 2.2. The proof of Theorem 2.1 below implies that, if $\mu_{11} = \mu_{22}$ and $\mu_{12} = \mu_{21}$, an X-model under FIFO has the same finite dimensional distributions as an $M/G/2$ system with service rate of each server equal to $\bar{\mu} = (\lambda_1 + \lambda_2)\theta(\mu, \lambda)/2$. Since an $M/G/2$ system can only be stable if $2\bar{\mu} > (\lambda_2 + \lambda_2)$, this proves Theorem 2.1 in this special case.